Homomorphic Encryption as a Solution of Trust Issues in Cloud

Abdellah EZZATI, Khalid EL MAKKAOUI*
LAVETE laboratory, Mathematics and Computer Science Department, Sciences and Techniques Faculty, Hassan 1er University
Settat, Morocco
abdezzati@gmail.com, kh.elmakkaoui@gmail.com

Abderrahim BENI HSSANE
LAROSERI laboratory, Computer Science Department
Sciences Faculty, Chouaïb Doukkali University
El Jadida, Morocco
abenihssane@yahoo.fr

***Abstract*—With the emergence of cloud computing, the concept of trust has become a major issue. Indeed, the key challenge is to ensure to customers that the selected cloud provider may store and process the raw data safely. If this is a storage service, data can be encrypted before sending them to the cloud server; in this case, data confidentiality is assured. However, before performing treatments, these data must be decrypted. It is this step that can be considered a breach of security. Indeed, the fear of seeing sensitive data be processed in crude is a major obstacle in adopting cloud services. To overcome this obstacle and strengthen confidence in the cloud services, in this article we will propose the adoption of Homomorphic Encryption methods that are able to perform operations on encrypted data without knowing the key secret.**

Keywords—Cloud Computing; Security; Trust; Confidentiality; Homomorphic Encryption.

I. INTRODUCTION

Cloud computing is becoming more and more a magic solution, thanks to the gains it presents at the level of cost of software, maintenance computer park and servers maintenance. However, safety concerns, including the fear of seeing confidential information processed in plain, is usually the main obstacle to the adoption of cloud services.

In this article, we will propose to the cloud providers using the homomorphic encryption technique to ensure the confidentiality of confidential data storage and processing in order to overcome the problem with the confidentiality of information and to build confidence in the adoption of cloud services.

The principle of this technique is to encrypt data before sending them to the cloud provider, which allows to perform encrypted data operations without having the secret key, and return a result that is the same as if we had worked directly on the raw data.

The rest of this paper is organized as follows: In Section II, we willl define cloud computing, we will present its service and deployment models, and its essential characteristics. In Section III, we will illustrate some applications of the technique of homomorphic encryption in various areas of the real world. In Section IV, we will define homomorphic encryption and present its operation and the categories that compose it. In Section V, we will illustrate the issues, confidence levels and the value of using homomorphic encryption techniques. In Section VI, we will illustrate the limits of HE cryptosystems. Finally, we will finish with section VII, in which we will present our conclusions and future work.

II. CLOUD COMPUTING

A. Definition:

According to a definition given by the NIST (US National of Standards and Technology), "Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction"[1].

This cloud model is composed of three service models (Software as a Service, Platform as a Service and Infrastructure as a Service) and four deployment models (Private, Public, Community and Hybrid Cloud) [1].

B. Essential characteristics:

Cloud computing services have characteristics that distinguish them from other technologies. The main characteristics are:

- *On-demand self-service:* Ability to provide a computing resource automatically without requiring human interaction on the side of the provider [1].
- *Broad network access:* Services are available on the network and accessible through standard mechanisms that promote use of client platforms (eg, mobile phones, tablets, laptops and desktops) [1].
- *Resource pooling:* The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at

a higher level of abstraction (e.g., country, state, or datacenter) [1].

- *Rapid elasticity:* Possibility to change very quickly the capacity provided, either more or less [1].
- *Measured service:* Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized service [1].

III. RELATED WORKS

Applications using the homomorphic encryption technique in the real world are very diverse and numerous. We will present here some.

The cloud private system of storage of electronic medical records of patients, has been proposed. In this system, all data in these files are encrypted by health care providers, before being transferred to the cloud storage system. Secret keys for access to the raw data folders are shared between the patient and the specific suppliers. However, this system does not allow the cloud to perform processing on the data without search by keywords. With the implementation of fully homomorphic encryption, cloud allows to perform operations on encrypted data, and send patient updates, alerts and relevant information based on the received data [2].

In the financial sector, there is a potential application scenario, with the objective of safeguarding confidential data and business customers and functions calculated on the data. Dissemination of relevant information such as data on companies, the stock price etc., are essential in making investment decisions. These data should be disse possible. The functions which make calculations on these data must be owners. They are based on new predictive models of the performance of share prices, which can be the result of costly research carried out by financial analysts. Most companies want to hide these private models to their competitors in order to preserve their investments. With the use of fully homomorphic encryption, some of these functions will be evaluated in private mode. The client will thus transfer an encrypted version of the function to the cloud, for example a program where some of the evaluations are specified encrypted entries. Streaming data is encrypted by the client's public key before being transferred to the cloud. Then the cloud service evaluates the private function on encrypted inputs using the encrypted program description. After the performance of operations on these data, cloud returns a result itself encrypted to the client [2].

Also, Sutar and Patil proposed an authentication framework in the Cloud, considering three parts, namely: Server Cloud, the cloud user and third parties. Then, using homomorphic encryption mechanism, the exchange of information between the parties mentioned will be preserved. Here, the authentication process is carried out by a third party after comparing the information of two other parties (user and the server cloud). This system has reduced the cloud server calculations and at the same time has preserved the confidential information of the user [3].

IV. HOMOMORPHIC ENCRYPTION

A. History

In 1978, Ronald Rivest, Leonard Adleman and Michael Dertouzos suggested for the first time homomorphic encryption concept [4]. RSA is a public key cryptosystem, which is a multiplicative homomorphic encryption system. The Shafi Goldwasser and Silvio Micali (GM) encryption system was proposed in 1982, it was an additive homomorphic encryption, but it can encrypt just a single bit [5]. In 1984, Taher ElGamal proposed a public-key cryptosystem, which is a multiplicative homomorphic encryption system [6]. In 1999, the French mathematician, Pascal Pailler proposed a new encryption algorithm, named cryptosystem Pailler, who was also an additive homomorphic encryption system [7]. In 2005, Dan Boneh, Eu-Jin Goh and Kobi Nissim invented an encryption system (BGN), with which we can perform an unlimited number of additions, but with only one multiplication [8]. In 2006, Xing Guangli et al have proposed a homomorphic encryption scheme which is extended to real numbers. In this system, the operations of addition, subtraction, multiplication and division are possible [9]. In 2008, Chen Liang and Chengmin Gao proposed Algebra Homomorphic Encryption Scheme Based On Updated ElGamal (AHEE) [10, 11]. In 2009, Craig Gentry implemented the fully homomorphic encryption scheme that was able to make many additions and multiplications using ideal lattices and with the bootstrap method [12]. In 2013, Gorti VNKV Subba Rao proposed Enhanced homomorphic Encryption Scheme (EHES) for homomorphic encryption / decryption with the IND-CCA secure system. This system allows you to perform operations of addition, multiplication and mixed operations [13].

B. Homomorphic Encryption (HE)

Homomorphic encryption systems are capable of performing operations on encrypted data without knowing the secret key. These operations generate a result, which is itself encrypted (i.e. incomprehensible even to cloud provider). The result obtained is the same as if we performed these operations on the raw data [14].

Mathematically speaking, we say that a system is homomorphic encryption if: from Enc(x) and Enc(y), it is possible to calculate Enc(f(x, y)), where f can be : +, ×, $\oplus$ [15].

The principles of the operation of the Homomorphic Encryption are as follows, and as shown in Figure 3[10, 14]:

1) **Key generation**: The client generates the public key
2) (**pk**) and the secret key (**sk**).
3) **Encryption**: The client encrypts the data with **pk**. And sends the encrypted data and **pk** to the Cloud server.
4) **Storage**: The encrypted data and **pk** are stored in the cloud database.

5) **Request**: The client sends a request to the server to perform operations on encrypted data.
6) **Evaluation**: The processing server processes the request and performs the operations requested by the client.
7) **Response**: Cloud provider returns to the client the processed result.
8) **Decryption**: The client decrypts the returned result, using **sk**.

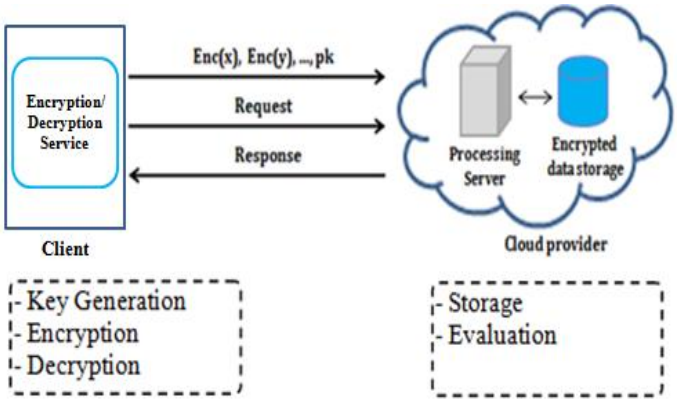

Figure 1. Homomorphic Encryption functions

Among the homomorphic encryption systems are distinguished, depending on the operations that evaluates raw data, multiplicative homomorphic encryption and additive homomorphic encryption.

- *Multiplicative Homomorphic Encryption:* A homomorphic encryption is multiplicative, if there is an algorithm that can calculate Enc (x × y) from Enc (x) and Enc (y) without knowing x and y [16].
- *Additive Homomorphic Encryption:* A homomorphic encryption is additive, if there is an algorithm that can calculate Enc (x + y) from Enc (x) and Enc (y) without knowing x and y [16].

Among Homomorphic Encryption systems, we distinguish three categories, depending on the operations performed on the data :

- **Partially Homomorphic Encryption (PHE) :** allows to perform operations on encrypted data, let multiplication or addition, but not both [17].
- **Somewhat Homomorphic Encryption (SWHE) :** allows to perform more than one operation, but a limited number of multiplication and addition operations [17].
- **Fully Homomorphic Encryption (FHE):** This is a cryptographic system that supports an unlimited number of both additions and multiplications [17].

Table I presents the categories different homomorphic encryption systems.

TABLE I. CATEGORIES OF HE SCHEMES

HE scheme	*PHE*	*SWHE*	*FHE*
RSA	✓		
GM	✓		
ElGamal	✓		
Pailler	✓		
BGN		✓	
AHEE			✓
Graig's			✓
EHFS			✓

C. *HE cryptosystems:*

HE systems are numerous. We present two cryptosystems are: ElGamal and EHES.

1) *ElGamal cryptosystem*

Preparation of Key
Input: $\mathbf{p} \in \mathbb{P}$ (random and large prime)
• Find a generator g of the multiplicative group Z^*_p • Choose a random integer $\mathbf{a} \in [2, p-2]$ • Compute $\mathbf{y} = g^a \bmod p$
Output: (pk, sk) public key: **pk** = (p, g, y) secret key: **sk** = (a)

Encryption: Enc (m, pk)
Input: message $\mathbf{m} \in Z_p$
• Choose a random integer $\mathbf{k} \in [2, p-2]$ • Compute $\mathbf{c_1} \equiv g^k \bmod p$ $\mathbf{c_2} \equiv m \times y^k \bmod p$
Output: encrypted message $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$

Decryption: Dec(c, sk)
Input: $\mathbf{c} = (\mathbf{c_1}, \mathbf{c_2})$
Compute $\mathbf{m} \equiv c_1^{-a} \times c_2 \bmod p$
Output: clear message $\mathbf{m} \in Z_p$

Figure 2. ElGamal Algorithm

Suppose we have two encrypted messages C1 and C2 by the ElGamal algorithm, such as: $C_1=(c_{11}, c_{12})$ et $C_2=(c_{21}, c_{22})$

Multiplicative:

$$\begin{aligned}(c_{11}, c_{12}).(c_{21}, c_{22}) &\equiv (c_{11}c_{21}, c_{12}c_{22}) \\ &\equiv (g^{k1}g^{k2}, (m_1 \times y^{k1})(m_2 \times y^{k2}) \bmod p \\ &\equiv (g^{k1+k2}, (m_1 \times m_2)y^{k1+k2}) \bmod p \\ &\equiv \text{Enc}(m_1 \times m_2, pk)\end{aligned}$$

2) EHES cryptosystem

Preparation of Key
Input: $p, q \in \mathbb{P}$ (large prime numbers) , such that $q < p$
Compute : $n = p \times q$
Output: (pk, sk) public key : pk = (n) Secret key : sk = (p, q)

Encryption : Enc (m, pk, sk)
Input: $m \in Z_p$
• Generate a random number **r** • Compute : $c \equiv m + r \times p^q \pmod n$
Output: $c \in Z_n$

Decryption : Dec (c, sk)
Input: $c \in Z_n$
Compute : $m \equiv c \bmod p$
Output: $m \in Z_p$

Figure 3. EHES Algorithm

Let x, y $\in Z_p$, pk = (n) and sk =(p, q)

Multiplicative:

$$Enc(x \times y) \equiv (Enc(x) \times Enc(y)) \pmod n, \text{ or}$$
$$x \times y = Dec(Enc(x) \times Enc(y)) \equiv (Enc(x) \times Enc(y)) \pmod p$$

Additive:

$$Enc(x + y) \equiv Enc(x) + Enc(y) \pmod n, \text{ or}$$
$$x + y = Dec(Enc(x) + Enc(y)) \equiv (Enc(x) + Enc(y)) \pmod p$$

V. TRUST ISSUES IN CLOUD COMPUTING

With the services offered by the cloud providers, companies can increase their productivity in the shortest possible time, with fewer staff and reduced costs. However, the adoption of such a service can only be done if security is ensured. Indeed, the major challenge is to strengthen the trust of customers by assuring them that the cloud providers may store and process data securely [18].

Indeed, ensuring optimum level of security has become a necessity to preserve the integrity, confidentiality and availability of services associated with the Cloud. for strengthen the trust of customers. We will classify this trust into two levels, according to the requirements of customers (businesses, consumers, etc.):

A. Level one of the trust:

At this level, customers can trust the cloud service providers if privacy, data integrity and service availability are ensured. The following figure shows level one of the trust.

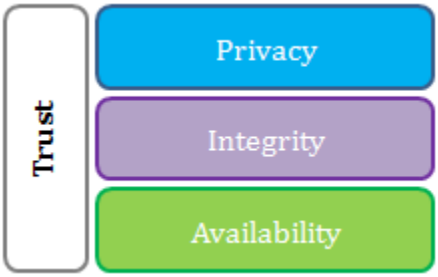

Figure 4. Level one of the trust

- *Availability:* is the property of information to be accessible and usable upon demand by an authorized entity[19].
- *Integrity:* is the property of information not to be altered. This means that the system must prevent undue modification of information (i.e, of a modification by unauthorized users or incorrect modification by authorized users) [20].
- *Privacy:* refers to the will of a user to control the disclosure of private information (authentication, authorization and access control) communication of encrypted data, and management of user identity [21].

B. Level two of the trust:

Even security associated with level one of the trust is assured, in the case of confidential data, customers require to ensure the confidentiality of storing and processing data. Figure 5 presents level two of the trust.

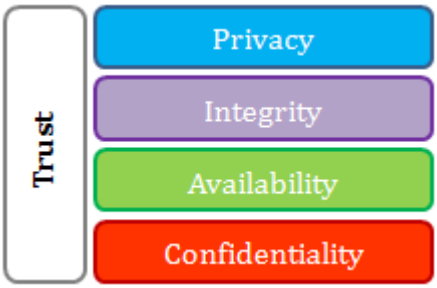

Figure 5. Level two of the trust

- *Confidentiality:* ensures that data remains confidential and invisible to the cloud provider, and even if the provider data centers have been attacked, customer data can neither be stolen nor reused [14].

In an unreliable environment , like in the public cloud , the confidentiality of the storage of confidential data and their treatment must be ensured. Thus, researchers noted a useful encryption method in this type of environment: homomorphic

encryption. Homomorphic encryption methods are able to perform operations on encrypted data without decrypting them and to give us results that are the same as if we had performed these operations on the raw data. This would allow us to outsource the calculation and storage of confidential data to the cloud, while keeping the secret keys that are essential to decrypting the results of operations performed on encrypted data.

VI. LIMITS OF HE CRYPTOSYSTEMS

Today, HE technique appears as the most effective and the safest for outsourcing the calculation and storage of confidential data to the cloud. However, HE systems have certain limitations. In the following we will present the limits of the ElGamal and EHES algorithms.

A. Limits of ElGamal and EHES

The table below presents the limits of cryptosystems: ElGamal and EHES.

TABLE II. LIMITS OF ELGAMAL AND EHES

Cryptosystem	Limits
ElGamal	pk=(p, g, y), sk=(a), et $m_1, m_2,..., m_k \in \mathbb{Z}_p$ **Multiplicative is true iff:** $\prod m_i < p$, with i={1,2,...,k}.
EHES	pk=(n), sk=(d), et $x_1, x_2,..., x_k \in \mathbb{Z}_p$ **Multiplicative is true iff:** $\prod x_i < p$, with i={1,2,...,k}. **Additive is true iff:** $\sum x_i < p$, with i={1,2,...,k}.

B. Exception of EHES

In some cases the encrypted message by EHES algorithm is itself that the clear message.

As indicated in Figure 3, the EHES encryption algorithm is as follows:

Enc(x) = x + r×p^q (mod n), with **pk**=(n) and **sk**=(p, q).

if **r×p^q = α×n**, with α ∈ ℕ* therefore, **Enc(x) = x**

Demonstration:

if $r \times p^q = \alpha \times n = \alpha \times pq \Leftrightarrow r \times p^{q-1} = \alpha \times q$
we get: p, q ∈ ℙ so now, q divides r.
Therefore, ∃ r ∈ ℕ* such that **r×p^q = α×n**

C. Enhanced EHES:

We improved EHES encryption algorithm, in in a way that the encrypted message is always different from the clear message. The new encryption algorithm of Enhanced EHES is as followings:

$$\mathbf{Enc(x) = x + p^{rq} \pmod n}$$

Demonstration:

If $p^{rq} = \alpha \times n = \alpha \times pq \Leftrightarrow p^{r(q-1)} = \alpha \times q$
we get: p, q ∈ ℙ so now, ∄ α ∈ ℕ* such that $\mathbf{p^{rq} = \alpha \times n}$

Therefore, always: **Enc(x) ≠ x**

VII. CONCLUSION AND FUTURE WORKS

In this article, we discussed the importance of adopting the homomorphic encryption technology for cloud providers. This technique allows them to preserve the confidentiality of sensitive data in order to strengthen the trust of their clients. Also, we have presented the limits of the ElGamal and EHES cryptosystems and we proposed an improved version of the EHES algorithm.

In our future work, we will focus on the analysis and improvement of homomorphic encryption algorithms of different systems, and we will determine their limits and performance.

ACKNOWLEDGMENT

I would like to thank my supervisor Mr. Abdellah EZZATI and my framing Mr. Abderrahim BENI HSSANE for the help, encouragement and guidance they have given me, in order to achieve this modest work.

REFERENCES

[1] P. Mell, T. Grance, "The NIST Definition of Cloud Computing," National Institute of Standards and Technology, U. S. Department of Commerce, September 2011.

[2] M. Ogburn, C. Turner, P. Dahal, "Homomorphic Encryption," In Complex Adaptive Systems, Publication 3, Cihan H. Dagli, Editor in Chief Conference Organized by Missouri University of Science and Technology 2013 - Baltimore, MD, Elsevier, 2013, pp. 502 – 509.

[3] S. Sutar, G. Patil, "Privacy Management in Cloud by making use of homomorphic Functions," International Journal of Computer Applications, 2012, 37(2), pp. 13 – 16.

[4] Rivest, Ronald L., Len Adleman, and Michael L. Dertouzos, "On data banks and privacy homomorphisms," Foundations of secure computation 4, N°. 11 (1978), pp 169 – 180.

[5] S. Goldwasser, S. Micali, "Probabilistic encryption and how to play mental poker keeping secret all partial information," in Proceedings of 14th Symposium on Theory of Computing , 1982, pp. 365 – 377

[6] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," IEEE Transactions on Information Theory, 1985, pp. 469 – 472.

[7] Pascal Paillier, "Public-key cryptosystems based on composite degree residuosity classes," In 18th Annual Eurocrypt Conference (EUROCRYPT'99), Prague, Czech Republic , volume 1592, 1999.

[8] D. Boneh, E. Goh, K. Nissim, "Evaluating 2-DNF formulas on ciphertexts," In Theory of Cryptography Conference, TCC'05, pp. 325–341,Springer, 2005.

[9] Xing Guangli, CHEN Xinmeng, ZHU Ping, MA Jie. "A method of homomorphic encryption," Wuhan University Journal of Natural Sciences, Vol.11, No.1, pp.181-184, 2006.

[10] Payal V. Parmar, Shraddha B. Padhar, Shafika N. Patel, Niyatee I. Bhatt, Rutvij H. Jhaveri, "Survey of Various Homomorphic Encryption algorithms and Schemes," In International Journal of Computer Applications (0975 - 8887), Volume 91 -No. 8, April 2014, pp. 26 – 32

[11] Chen, Liang and Chengmin Gao, "Public Key Homomorphism Based on Modified ElGamal in Real Domain," In International Conference on Computer Science and Software Engineering, Vol. 3, pp.802-805. 2008.

[12] Graig Gentry, "A fully homomorphic encryption scheme," Ph.D. dissertration, Stanford University, 2009. http://crypto.stanford.edu/craig/craig-thesis.pdf.

[13] G. VNKV Subba Rao, Md.Sameeruddhin Khan, A.Yashwanth Reddy, K.Narayana, "Data Security in Bioinformatics," In International Journal of Advanced Research in Computer Science and Software Engineering 3(11), November - 2013, pp. 590 – 598.

[14] M. TEBAA, S. EL HAJJI, A. EL GHAZI, "Homomorphic Encryption Applied to the Cloud Computing Security," In Proceedings of the World Congress on Engineering 2012 Vol I, WCE 2012, July 4 - 6, 2012 London, U.K.

[15] Ronald L. Rivest, Leonard Adleman, and Michael L. Dertouzos, "On Data Banks and Privacy Homomorphisms", chapter On Data Banks and Privacy Homomorphisms, pages 169-180. Academic Press, 1978.

[16] Xing Guangli, CHEN Xinmeng, ZHU Ping, MA Jie. "A method of homomorphic encryption," Wuhan University Journal of Natural Sciences, Vol.11, No.1, pp.181-184, 2006.

[17] M. Ogburn, C. Turner, P. Dahal, "Homomorphic Encryption," In Complex Adaptive Systems, Publication 3, Cihan H. Dagli, Editor in Chief Conference Organized by Missouri University of Science and Technology 2013 - Baltimore, MD, Elsevier, 2013, pp. 502 – 509.

[18] Reem Alattas, "Cloud Computing Algebraic Homomorphic Encryption Scheme," In International Journal of Innovation and Scientific Research, ISSN 2351-8014 Vol. 8 No. 2 Sep. 2014, pp. 191-195.

[19] Cloud Select Industry Group (C-SIG) of European Union, " Cloud Service Level Agreement Standardisation Guidelines," June 2014.

[20] Mazhar Ali, Samee U. Khan , Athanasios V. Vasilakos, "Security in cloud computing: Opportunities and challenges," Elsevier, 2015.

[21] Sweta Agrawal and Aakanksha Choubey, "Survey of Fully Homomorphic Encryption and Its Potential to Cloud Computing Security," In International Journal of Advanced Research in Computer Science and Software Engineering, Volume 4, Issue 7, July 2014.